I Claim:

1. A method for detecting cross-iteration dependencies between variables in a loop of a computer program, the method comprising the steps of:

associating unique values with each of the values of indirect loop index variables of the loop;

calculating for each iteration of the loop an indirectly indexed access pattern based upon the associated unique values; and

determining whether cross-iteration dependencies exist between any two iterations of the loop based upon the indirectly indexed access pattern of the two iterations.

2. The method as claimed in claim 1, wherein the unique values associated with each of the values of the indirect loop index variables of the loop are different binary bit patterns of a bit vector.

3. The method as claimed in claim 2, wherein the indirectly indexed access pattern for an iteration is calculated by forming the logical AND of the unique bit patterns associated with each of the values of the indirect loop index variables of the loop for that iteration.

4. The method as claimed in claim 3, wherein the existence of cross-iteration dependencies is determined by determining whether the indirectly indexed access pattern for the two iterations have any common bit positions that share a value of one.

5. The method as claimed in claim 1, wherein the unique values associated with each of the values of the indirect loop index variables of the loop are different prime numbers.

6. The method as claimed in claim 5, wherein the indirectly indexed access pattern for an iteration is calculated by forming the product of the unique prime numbers associated with each of the values of the indirect loop index variables of the loop for that iteration.

7. The method as claimed in claim 6, wherein the existence of cross-iteration dependencies is determined by determining whether a greatest common divisor between the two indirectly indexed access patterns for the two corresponding iterations is greater than one.

8. The method as claimed in claim 1, further comprising the step of grouping iterations in a wave for execution in a common time period such that no cross-iteration dependencies exist between any of the grouped iterations of the wave.

9. The method as claimed in claim 8, further comprising the step of executing each of said waves in a prescribed sequence, and executing each of said iterations in each of said waves in parallel with each other.

10. A method for assisting in scheduling parallel computation of instructions in a loop of a computer program, the method comprising the steps of:

determining, for a loop, active array variables, and direct and indirect loop index variables;

determining, for each iteration of the loop, values of the indirect loop index variables;

associating a unique value with each of values of the indirect loop index variables;

calculating an indirectly indexed access pattern for each iteration of the loop; and

determining whether cross-iteration dependencies exist between any two iterations of the loop based upon the indirectly indexed access patterns of the two iterations.

11. A method for detecting cross-iteration dependencies between variables in a loop of a computer program, the method comprising the steps of:

determining, for a loop, Boolean conditions embedded in the loop;

determining possible decision paths an iteration is allowed to take in the loop body in the presence of Boolean conditions;

determining a first set $V = \{v_1, v_2, \ldots v_n\}$ of active array variables of the loop, and a second set $I = \{i_1, i_2, \ldots i_r\}$ of indirect loop index variables that appear with the active array variables of said first set in the loop;

determining, for each decision path λ in the set of possible decision paths in the loop, the set V_λ of active array variables associated with the path, and the set I_λ of indirect loop index variables that appear with the active array variables V_λ.;

determining the range $[N_1, N_2]$ of the direct loop index i of the loop, and the maximal range $[M_1, M_2]$ of the set of indirect loop index variables $i_1, i_2, \ldots, i_r$ of the loop;

associating, for each value l of the indirect loop index variables in the range $[M_1, M_2]$, a unique value $p(l)$ with each value l of the indirect loop index variables in the range $[M_1, M_2]$;

determining, for each pair (i, λ) of a value of the direct loop index i and a decision path λ in the set of all possible decision paths in the loop, the value of $S_\lambda(i)$ where $S_\lambda(i)$ is the product of the unique values $p(l)$ associated with each value l of the indirect loop index variables in the range $[M_1, M_2]$; and

determining, for any pair $S_\alpha(i)$ and $S_\beta(j)$, whether the values of $S_\alpha(i)$ and $S_\beta(j)$ indicate that cross-iteration dependencies exist between iterations i and j.

12. The method as claimed in claim 11, wherein the unique values $p(l)$ are different prime numbers and the values of $S_\alpha(i)$ and $S_\beta(j)$ indicate that no cross-iteration dependencies exist between iterations i and j if the greatest common divisor of $S_\alpha(i)$ and $S_\beta(j)$ is 1 when $i \neq j$.

13. The method as claimed in claim 11, wherein the unique values $p(l)$ are different bit patterns in a bit vector, and the values of $S_\alpha(i)$ and $S_\beta(j)$ indicate that no cross-iteration dependencies exist between iterations i and j if $S_\alpha(i)$ and $S_\beta(j)$ do not share any common bits that are set to 1 when $i \neq j$.

14. The method as claimed in claim 11, further comprising the step of:

grouping loop iterations in waves such that any two iterations i and j of a particular wave have no cross-dependencies.

15. The method as claimed in claim 14, further comprising the step of:

executing the waves in which loop iterations are grouped in a predetermined sequence such that the wave having the lowest value of the direct loop index i is executed first and completely, followed by the wave with the next lowest value of i, and so on for successive values of i; and

16. The method as claimed in claim 14, further comprising the step of:

executing the iteration in each wave in parallel using multiple computing processors.

17. A computer program product for detecting cross-iteration dependencies between variables in a loop of a computer program, the computer program product comprising computer software stored on a computer-readable medium for performing the steps of:

associating unique values with each of the values of indirect loop index variables of the loop;

calculating for each iteration of the loop an indirectly indexed access pattern based upon the associated unique values; and

determining whether cross-iteration dependencies exist between any two iterations of the loop based upon the indirectly indexed access pattern of the two iterations.

18. A computer system for detecting cross-iteration dependencies between variables in a loop of a computer program, the computer system executing computer software stored on a computer-readable medium for performing the steps of:

associating unique values with each of the values of indirect loop index variables of the loop;

calculating for each iteration of the loop an indirectly indexed access pattern based upon the associated unique values; and

determining whether cross-iteration dependencies exist between any two iterations of the loop based upon the indirectly indexed access pattern of the two iterations.